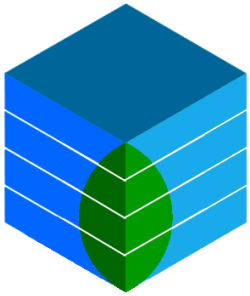

BigStorage

STORAGE-BASED CONVERGENCE BETWEEN HPC AND CLOUD TO HANDLE BIG DATA

Deliverable number	D4.1
Deliverable title	Intermediate Report WP4 Storage Solutions
Editor	Angelos Bilas (FORTH)
Main Authors	Alvaro Brandon (UPM), Rizkallah Touma (BSC), Athanasios Kiatipis (Fujitsu), Fotis Nikolaidis (CEA), Dimitrios Ganosis (FORTH), Georgios Koloventzos (BSC), and Michał Zasadziński (CA)

Grant Agreement number	642963
Project ref. no	MSCA-ITN-2014-ETN-642963
Project acronym	BigStorage
Project full name	BigStorage: Storage-based convergence between HPC and Cloud to handle Big Data
Starting date (dur.)	1/1/2015 (48 months)
Ending date	31/12/2018
Project website	http://www.bigstorage-project.eu

Coordinator	María S. Pérez
Address	Campus de Montegancedo sn. 28660 Boadilla del Monte, Madrid, Spain
Reply to	mperez@fi.upm.es
Phone	+34-91-336-7380

Executive Summary

This document provides an overview of the progress of the work done until M24 of the Project BigStorage (from 01-01-2015 until 31-12-2016) in WP4 Storage Solutions.

The tasks in WP4 cover problems related to device technology and how these affect storage architecture and the storage stack. WP4 categorizes issues and tasks as:

T4.1 Storage acceleration: How modern storage systems can take advantage of heterogeneity and locality to improve performance of applications.

T4.2 Storage convergence: How storage systems can support different types of applications, especially where big data is involved, converging different uses of storage over the same platforms.

T4.3 Storage isolation: How we can infer system behavior in mixed environments, where there is heterogeneity in terms of storage technologies but also the applications that use storage.

The work carried out by ESRs and in relation to WP4 covers these topics as follows:

- Prefetching mechanisms and polices that have the potential to deal with heterogeneity.
- File systems for heterogeneous storage systems.
- Support for big data – big storage applications.
- Cloud blobs as a basis for supporting diverse applications.
- Multi-criteria optimization for dealing with performance issues.
- Monitoring and inference for understanding the impact of co-location.
- Root cause analysis for detecting causality relationships in complex environments.

This report presents an overview of each of these topics, including a brief analysis of the state of the art in each case and a preliminary plan of the specific problems to be addressed by each ESR.

Document Information

IST Project Number	MSCA-ITN-2014-ETN-642963
Acronym	BigStorage
Title	Storage-based convergence between HPC and Cloud to handle Big Data
Project URL	http://www.bigstorage-project.eu
Document URL	http://www.bigstorage-project.eu
EU Project Officer	http://bigstorage-project.eu/index.php/deliverables
Deliverable	D4.1 Intermediate Report on WP4
Workpackage	WP4 Storage Solutions
Date of Delivery	Planned: 31.12.2016 Actual: 20.12.2016
Status	Version 1.0 final ■ draft □
Nature	prototype □ report ■ dissemination □
Dissemination level	public □ consortium ■
Distribution List	Consortium Partners
Document Location	<url of the document>
Responsible Editor	Angelos Bilas (FORTH), bilas@ics.forth.gr, Tel: +302810391669
Authors (Partner)	Alvaro Brandon (UPM), Rizkallah Touma (BSC), Athanasios Kiatipis (Fujitsu), Fotis Nikolaidis (CEA), Dimitrios Ganosis (FORTH), Georgios Koloventzos (BSC), and Michał Zasadziński (CA)
Reviewers	All advisors
Abstract (for dissemination)	The WP4 covers problems related to device technology and how these affect storage architecture and the storage stack. WP4 categorizes issues and tasks as: **T4.1 Storage acceleration:** How modern storage systems can take advantage of heterogeneity and locality to improve performance of applications. **T4.2 Storage convergence:** How storage systems can support different types of applications, especially where big data is involved, converging different uses of storage over the same platforms. **T4.3 Storage isolation:** How we can infer system behavior in mixed environments, where there is heterogeneity in terms of storage technologies but also the applications that use storage. This report presents an overview of each of these topics, including a brief analysis of the state of the art in each case and a preliminary plan of the specific problems to be addressed by each Early Stage Researcher (ESR).
Keywords	Storage acceleration, storage convergence, storage isolation, heterogeneity, Flash, NVM, monitoring, inference

Version	Modification(s)	Date	Author(s)
0.1	Initial template and structure	22.09.2016	Angelos Bilas, FORTH
0.2	Sections from all authors	14.10.2016	All authors
0.3	Internal version for review	28.10.2016	Angelos Bilas, FORTH
0.4	Comments from internal reviewers	11.11.2016	Internal reviewers
0.5	Final version to commission	30.11.2016	Angelos Bilas, FORTH

Project Consortium Information

Participants		Contact
Universidad Politécnica de Madrid (UPM), Spain		María S. Pérez Email: mperez@fi.upm.es
Barcelona Supercomputing Center (BSC), Spain		Toni Cortes Email: toni.cortes@bsc.es
Johannes Gutenberg University (JGU) Mainz, Germany		André Brinkmann Email: brinkman@uni-mainz.de
Inria, France		Gabriel Antoniu Email: gabriel.antoniu@inria.fr Adrian Lebre Email: adrien.lebre@inria.fr
Foundation for Research and Technology - Hellas (FORTH), Greece		Angelos Bilas Email: bilas@ics.forth.gr
Seagate, UK		Malcolm Muggeridge Email: malcolm.muggeridge@seagate.com
DKRZ, Germany		Thomas Ludwig Email: ludwig@dkrz.de
CA Technologies Development Spain (CA), Spain		Victor Muntes Email: Victor.Muntes@ca.com
CEA, France		Jacque Charles Lafoucriere Email: Charles.LAFOUCRIERE@CEA.FR
Fujitsu Technology Solutions GMBH, Germany		Sepp Stieger Email: sepp.stieger@ts.fujitsu.com

Table of Contents

1 Introduction

1.1 *WP4 Overview*

This subsection discusses an overview of WP4 as was presented in the proposal.

Storage systems are undergoing fundamental changes to keep up with the requirements of new applications and services. At the infrastructure level the problems we face today can be categorized in the areas storage acceleration, storage convergence, and storage isolation:

T4.1 Storage acceleration received much attention, largely due to the advent of Flash-based storage device technologies. Future disruptive change to the I/O hierarchy will be from the use of emerging, byte addressable Non-volatile Memories (NVM), which are candidates for bridging the gap between non-persistent byte-addressable DRAM and persistent, block-addressable storage.

T4.2 Storage convergence refers to bringing storage and computation closer both in the data centre and HPC. Traditionally, persistent storage has been placed behind a storage area network (SAN) for scaling and management purposes. With current technology trends, it becomes important that storage (and NVM) moves closer to the compute nodes to further reduce energy footprint. This is a significant architectural shift and requires fundamentally different approaches to storing, caching, replicating, and moving data.

T4.3 Storage isolation is emerging as a central problem for storage infrastructures in heterogeneous and multi-tenant environments. Access to storage (and infrastructure in general) leads to contention that is induced even when independent applications access different parts of the data on the same storage devices, resulting in a loss of efficiency at the device and I/O path level. As a consequence, providers prefer to reduce shared access to storage resources by over-provisioning, leading to increased costs. This problem is particularly acute in the era of Big Data and Cloud Storage where data access is at the heart of applications and services.

WP4 will examine Big Data issues associated with deep storage hierarchies, how storage can be architected closer to computation, and how we can ensure application isolation over shared infrastructures.

1.2 *ESR Participation and contributions*

The full list of ESRs in the current form of the project is:

ESR	Name	Institution	Main advisor
1	Ovidiu Marcu	INRIA	Gabriel Antoniu/Alexandru Costan
2	*Alvaro Brandon*	*UPM*	*Maria S. Perez*
3	Pierre Matri	UPM	Maria S. Perez
4	Muhammad Umar Hameed	Mainz	Andre Brinkmann

5	*Rizkallah Touma*	*BSC*	*Anna Queralt/Toni Cortes*
6	Fotios Papaodyssefs	Seagate	Malcolm Muggeridge
7	Linh Thuy Nguyen	INRIA	Adrien Lebre
8	*Athanasios Kiatipis*	*Fujitsu*	*Sepp Stieger*
9	*Fotis Nikolaidis*	*CEA*	*Philippe Deniel*
10	*Dimitrios Ganosis*	*FORTH*	*Angelos Bilas/Manolis Marazakis*
11	*Nafiseh Moti*	*Mainz*	*Andre Brinkmann*
12	*Georgios Koloventzos*	*BSC*	*Ramon Nou/Toni Cortes*
13	Mohammed-Yacine Taleb	INRIA	Shadi Ibrahim
14	Yevhen Alforov	DKRZ	Thomas Ludwig / Michael Kuhn
15	*Michał Zasadziński*	*CA*	*Victor Muntes*

The ESRs that were planned to participate in WP4 and the respective tasks are:

- ESR8: T4.2
- ESR9: T4.2 T4.3
- ESR10: T4.1 T4.2 T4.3
- ESR11: T4.1 T4.2 T4.3
- ESR12: T4.1 T4.2 T4.3

In addition, the work of certain other ESRs has overlap with WP4 so we include that work in the report as well. These ESRs are: ESR2, ESR5, and ESR15. At the current stage of work the research by individual ESRs contributes to each task as follows:

T4.1 Storage acceleration: This work shows how modern storage systems can take advantage of heterogeneity and locality to improve performance of applications.

- ESR5: Prefetching
- ESR11: Joined the project in Sep. 2016 and there was not enough progress.
- ESR12: Heterogeneous file systems

T4.2 Storage convergence: This work shows how storage systems can support different types of applications, especially where big data is involved, converging different uses of storage over the same platforms.

- ESR8: Big data – big storage
- ESR9: Cloud blobs

T4.3 Storage isolation: This work shows how we can infer system behavior in mixed environments, where there is heterogeneity in terms of storage technologies but also the applications that use storage.

- ESR2: Multi-criteria optimization
- ESR10: Monitoring and inference
- ESR15: Root cause analysis

Next, we discuss each of these categories and the associated ESR work in mode detail.

2 Storage acceleration (T4.1)

2.1 *ESR5: Prefetching and Data Placement via Static Code Analysis*

Contributor: *Rizkallah Touma, BSC*

Overview

Prefetching can be defined as ``transferring data from persistent storage to main memory in anticipation of later use''. It is a common approach to improving access times to data and many different prefetching techniques have been proposed over the years.

An area where prefetching has been widely studied is Persistent Object Stores (POS). A POS is storage systems that exposes persistent data in the form of objects. The underlying storage mechanism of a POS varies from simple binary serialization of objects to complex object-oriented databases and Object-Relational Mapping systems (ORM). The most popular object-oriented databases include Caché, DB4O and Versant while the most used ORM systems are Hibernate, Apache OpenJPA and Data Nucleus.

Most approaches to prefetching have been based on heuristics and machine learning techniques that generate prefetching hints through monitoring runtime access to the store. This process adds a non-negligible overhead to application execution time and consumes a considerable amount of memory. Moreover, it might result in false positives that cause unnecessary movements of objects from persistent storage to main memory, thus harming the application's performance and limiting the benefits of prefetching.

We propose to use static code analysis of object-oriented applications to automatically generate prefetching hints. Our approach takes advantage of the symmetry between application objects and POS objects to perform the analysis and generation process at the application-level without involving the underlying POS. Furthermore, analyzing the application code allows us to generate accurate prefetching hints that do not contain false positives without any input from the developer. Finally, the approach is done prior to the execution of the application and has no effect on its performance.

Similarly, we can use static code analysis in order to generate information relevant to the data placement of objects in a distributed storage system. Knowing which objects are likely to be accessed together in the application allows us to apply smarter data placement strategies which in turn help save time and memory thus improving the performance of the application.

Gaps in state-of-the-art

The structure in which POSs expose data, in the form of objects and relations between these objects, is rich in semantics ideal for detecting access patterns [1] and has invited a significant amount of research on how to generate prefetching hints based on these patterns.

On the one hand, the automatic generation of prefetching hints is usually done while the application is being executed (e.g. [43], [44], [45]), which adds a non-negligible overhead to application execution time and consumes a considerable amount of memory. Moreover, the process might result in false positives that cause unnecessary movements of objects from persistent storage to main memory, thus harming the application's performance and limiting the benefits obtained from prefetching.

On the other hand, manually supplying prefetching hints is done by inspecting the application code prior to application execution. It generally results in more accurate hints given that the developer has better knowledge of the data accesses of the application but it is a tedious task that requires manual inspection of the entire application code. Moreover, correct prefetching hints are difficult to determine and maintain and incorrect ones are hard to detect [46].

Our approach aims to combine the advantages of the two types of techniques. We plan to use static code analysis to mimic the manual specification of prefetching hints, hence providing the same accuracy of manually supplied hints. At the same time, our approach is fully automatic and does not require any input from the developer. It is also done prior to application execution and does not have an effect on its performance.

Goals

- Defining a novel approach to automatically generate prefetching hints based on static code analysis. The approach is done prior to application execution and does not generate any false positives.
- Expanding the approach in order to generate hints for data placement policies depending on the information gathered during the static code analysis of the application.
- Extending the literature on static code analysis by using it for the new purpose of generating prefetching hints.

2.2 *ESR12: Heterogeneous File Systems*

Contributor: *Georgios Koloventzos, BSC*

Overview

Storage systems are about to make another leap in the near future: a new NVRAM technology is emerging and promises as fast I/O transactions as RAM modules but as a permanent storage system. On the other hand, HDDs are still in use as the main storage systems for most of the machines around the world, even though new approaches in HDDs gave more capacity, they are still bounded to their hardware problems in terms of I/O time. SSDs are reaching a level where we can obtain fast, cheap and adequate capacity. But recent researchers have found that using SSD as cache and at the end as main storage systems can wear the disk too much, resulting to HDD performance in the worst cases [47][48]. Such results are not good for the reliability and sustainability of the SSDs. More and more machines are focusing on having hybrid storage system infrastructure in order to tackle the problems of each systems'

drawbacks. File systems are not build to adjust to these new approaches and new technologies of storage systems.

The approach in building software also is not in line with the file systems [51]. Programmers' approach on atomicity and reliability of a program leads to big overhead with the storage and file systems. New file structures with tables, compressed data and mini databases are in contrast with what a file used to be. Instead of being a continuous string of data, files have become a miniature file system themselves. Also recent studies [49][50] have found that only a part of the libraries are used most of the time. The rest are creating a long tail that are used only a handful of times. Furthermore new libraries are developed to tackle programming approaches each operating system uses. Results like this should be taken into account in order to place libraries in particular storage systems for faster fetching to RAM. This can lead to faster boot and execution of programs that use these particular libraries.

Throughout the last years our assumptions in building software and hardware have changed. We have to mend new (and old) hardware with our programming principles and the operating systems. The response from the community is minimal and its mostly focus on one side each time. Our approach will take into account both problems and create a middle ground for both. Our work will offer a better approach of accessing a file using all the advantages of all modern and old storage systems by kernel. With such approach we will not force the user in new assumptions or changing their perception on how their systems should be build.

Gaps in state-of-the-art

Researchers are creating file systems implementations that are focused mainly on a particular storage system. SSDs have features, which are not taken into consideration from the operating systems. Thus most of these research projects are focusing to this particular storage system [6,7,8]. The same approach is taken also with NVRAM technology. It is considered as the medium between RAM and SSD. The first approaches in research are to test it as a cache layer for I/O. Furthermore file systems specific to this storage system are emerging.

In both situations we believe that it is not the correct approach the community should follow. The research is focused in specific problems on how to make the I/O to a particular storage system better. Taking a step backward we can see that as now each file structure has became a file system itself we should approach the I/O differently. No research has been done in combining all the storage systems as one. Exploit all the benefits of each one to create a true heterogeneous file system. The best result will be succeeded when each storage system will do what it is best for itself and for the group. ZFS has the advantage to create a pool of different parts of storage systems under the hood of a seamless file system. Thus the ZFS is the best candidate for a solution in the problem of creating a heterogeneous file system.

Current research projects do not take into consideration the file structure. Most of them are consist from smaller parts that can be distinguished and separated. A header of a file that is read once is different from a bitmap that have interaction in each request. Such differences can give us leverage to

associate parts of files with specific storage systems Also logging the requests we will may have distinguish patterns that at the end give us a boost in specific operations. Such as, if we can see what parts of file are needed for booting a machine, we can always keep them in the fastest storage to speed up boot time.

Goals

- Log I/O requests for analysis: Operating systems and programs are accessing files, libraries in different patterns. Such log can give us insight on how files are structured/accessed
- Extract patterns of read/write requests: Not all files are written and read the same way. Should files written in one system and read form another (probably libraries that are written once but accessed a lot)?
- Optimize where a file is stored in comparison with the pattern of its requests
- Track which parts of a file is requested more: Libraries have functions/ parts of files that are accessed on a regular basis or a bitmap of a database file.
- Optimize where parts of file are stored: Different file structures have different portion of file accessed in a different way. Headers are accessed sequentially rather than bitmaps are accessed bit per bit.

3 Storage Convergence (T4.2)

3.1 *ESR8: Big Storage Handling – Big Data Processing*

Contributor: *Athanasios Kiatipis, Fujitsu*

Overview

The objective of this job is to design next-generation data processing models for applications that require general data orchestration, independent of any programming model. Based on the Petabyte-Storage Device ETERNUS CD10000, the focus of the research is on workflows, which link data and applications on high scale petabyte dimensions. The CD10000 is a server-based product, which includes all hardware and software components, in order to provide software-defined storage (CEPH with Object Block File and S3 Interfaces).

Huge amounts of data require a change in the model of handling them. Previous technologies tend not to expand to the new dimensions. This opens ranges of opportunities to solve the existing issues (like virtualization, sync-and-sharing or mobile data) in new storage dimensions and new storage approaches. The examination on how old and new storage systems can interact and how storage clusters and computation can come work closer in future is also part of the project. The research focuses on the integration of new scale-out storage systems into new application areas. Therefore, these opportunities are based on a list of example applications. The resulting approaches should generalize the gained insights, helping to bring storage and computation closer together.

The current research specializes on the integration of Internet of Things data streaming into the CD10000 ETERNUS system. The Industrie4.0 concept is of vital importance for the future of production in Germany, bringing the opportunity of evaluating the CD10000 system for such cases. Also, backups of configurations could help with the migration of a whole cluster to another hardware, in case of a failure or if a similar setup is needed. Future research topics could also include the integration of machine learning techniques into data management.

Gaps in state-of-the-art

The CD10000 system is a new scale-out device that utilizes CEPH in order to manage vast amounts of data. As a newly released system, it has many aspects that can be tested and eventually improved:

- Multi-location for CD10000 cluster, in a distance of 40 km or below.
- The handling of very small files by CEPH, for IoT applications.
- Benchmarking the CD10000: Techniques and applications.
- Backup of management node configuration.

Goals

- Deepen knowledge on Software Defined Storage and especially CEPH.
- Execute performance tests on specific configuration of a CEPH cluster, by using the Fujitsu CD10000 system.
- Extend the knowledge of the company on possible configurations on stacked switches or storage nodes cabling.
- Examine the situation with the handling of small files, for Internet of Things applications, of the CD10000 system.
- Integration with the engineering developing team of CD10000 in order to gain valuable insights on the current problems/bugs and how can be improved.
- Perform benchmarks on multi-location applications.
- Examine the possible utilization of CD10000 on the use cases that will be examined in Big Storage project.
- Machine learning applications on data stored in CD10000.

3.2 *ESR9: Storage blobs hosted in the cloud*

Contributor: *Fotis Nikolaidis, CEA*

Overview

On modern big data world, traditional storage interfaces like filesystems have been proved insufficient due to the imposed common paths and global namespaces. To mitigate this issue object storage interfaces were conceived, where operations are limited to PUSH and GET, and data are accessible by unique ids. Using that, every data element (object) is inherently isolated from the rest, allowing dramatic increase on scalability.

However, once an object is PUSHed, the ownership is delegated to the storage provider while the actual data owner (client) rely on the promise that data will be available for later GET operations. In the meanwhile, the provider is free to manipulate data (e.g compress, replicate, strip, encode) at will. Given that, it is impossible for the client to span data across different providers unless it explicitly takes place on the application layer. If so, the application becomes tenacious and developer is required to have extended knowledge on storage systems.

A main goal of this project is to revise the existing status quo and propose a new flexible model that will benefit both the client and the storage administrator. Contrary to existing approaches where both the storage plane and the control plane are located within the provider, our model separates the two. The provider exposes resources located within its realm (storage plane), to an external manager (control plane).

The external manager in our system is called layout, and is a self-contained entity encapsulating pointers to resources, chunk distribution metadata and control routines. The latter is significantly important since

it enables the client to deploy custom functions on the control plane, something that is not feasible with existing technology. The purpose of a layout is to act as an entry point to conceptually related dataset (e.g a file, an account, an application) dispersed over multiple remote resources.

A layout is hosted on an intermediate service, between the application and the storage provider. If an application needs access to the data, it must go through that layout which acts as a request coordinator (Application to Service). This makes possible to enforce security policies on a dataset, support complex group-based sharing methods, support multiple interfaces for the same dataset, etc.

On the lower side of a layer (Service to Storage) we can benefit from having resources to different providers, which can lead to QOS policies, reduced cost, enhanced security and complex distribution schemas (backup, migration, striping, redundancy).

Gaps in state-of-the-art

Traditionally the storage provider is a black box to the client. In our model, the storage provider is simply a holder of exposed resources. This new approach combined with the fact that the data management is taken away from the storage provider, broadens the horizon of appliances. New features, non existed or minimally supported by current technology, are feasible.

Firstly, a client is able to perform complex QOS schemas since it's possible to choose the resources that better fit to a given role. For instance, the client may use SSD resources from provider A and HDD resources from provider B if some of the data are more latency-sensitive than the other.

Using a single entry to a dataset makes possible to have group-based views. While the dataset remains unchanged, each group (e.g admin, users, editors) will be provided with different views.

In traditional approaches, the group-based view is build on the application layer which poses a great security risk. On our approach, a layout acts as an entry point to the dataset therefore is easier to do security (or any policy) enforcement at that point without any changes to the applications.

Another security aspect is that of unauthorized access. Currently, data are trusted to a single provider. If there is a security breach or the provider is forced by the government, the data are exposed.

On the other hand in our architecture, data are dispersed over multiple resources of multiple providers. As a result, none of the storage providers has the full dataset. The only one entity which can reconstruct it is the layout which lives outside of the providers.

As for the scalability, the layouts (remember each layout is an isolated, self-contained entity) are scalable both vertically and horizontally. Vertically because a resource is a virtual entity with mutable capacity, and horizontally because new resources can be added if required.

Last but not least, existing applications are not required to be changed. The aforementioned client, is a library running to the same host as the application side and exposes conventional interfaces like filesystems and block devices. Applications use these interfaces and the library converts operations to the corresponding layout actions.

Goals

To summarize, the proposed architecture involves 3 parts: The client (library), the layout, and the storage provider. The client runs close to the application and acts as a translator between the interface and the layout. The layout, which runs as a remote service and is responsible for the data management of resources located over multiple storage providers. The benefits of this architecture are:

- Advanced data sharing (group-based context, data enrichment, etc.)
- Enforce security policies on that datapath instead of the application-level
- Advanced QOS schemas to better match the expected access pattern
- Data ownership is not delegated to the storage provider
- Prevention of third party unauthorized data access
- Both horizontal and vertical scalability
- Custom control routines on the datapath

4 Storage Isolation (T4.3)

4.1 *ESR2: Multi-criteria Decision Support Systems for Big Data Analytics*

Contributor: *Alvaro Brandon, UPM*

Overview

Big Data tools and infrastructures are complex system where different variables and their impact in performance have to be considered. Nowadays, this decision space can be overwhelming for users leading to poor choices and the need of expert knowledge when using data analytics platforms. The aim of this PhD is to build models that explain what variables to consider when using Big Data analytics including parameterization of the tools, system configuration, data placement and job allocation. To achieve this we will use machine learning techniques that can be easily trained and adapted to any environment

Gaps in state-of-the-art

The most relevant publications we found that optimise storage depending on computation are:

- MARLA: MapReduce for heterogeneous clusters [39]
- Scarlett : Coping with Skewed Content Popularity in MapReduce Clusters [40]
- Improving MapReduce performance through data placement in heterogeneous Hadoop clusters [41]

The main concern of these publications is heterogeneity and how it affects outliers and data location. The number of replicas stored by HDFS is agnostic of the computation power of the nodes. Faster nodes will process their local data quickly. This means that they will have to read data remotely from other machines to keep allocating tasks. This creates an unnecessary traffic across the network that can be a bottleneck. This is addressed in [40] by replicating based on popularity of the accessed blocks and in [41] by establishing a computing ratio for each node. MARLA [39] allows for worker nodes to request work as they complete tasks. In this way the nodes process their own fair share of work. It does so by sitting on top of a shared-file system.

Since we are going to use machine learning through monitoring the applications we can also take this kind of decisions. One of the questions we want to address is what storage technologies are more beneficial to which workloads. Also since new computing platforms such Spark caches data in memory, we want to address the importance of data locality, depending on the application.

Goals

- Extract statistical information about the workloads executed in one cluster and correlate their performances with the storage technologies and technologies available.

- Evaluate the impact of data locality depending on the type of application and use it in the decision making process when launching applications.
- Detect bottlenecks in I/O and configure applications accordingly to avoid these bottlenecks.

4.2 *ESR10: Storage optimization using emerging storage devices*

Contributor: *Dimitrios Ganosis, FORTH*

Overview

With the advent of Big Data as well as modern HPC challenges, storage is undergoing fundamental changes to keep up with demand and requirements of new applications and services. At the storage system level, problems faced today by infrastructure can be categorized in three broad areas: storage acceleration, storage convergence and storage isolation.

This work examines future storage system, that is heterogeneity and interference-aware, which leverages low-latency storage devices for improving the performance of Big Data applications. It will also provide a scalable, distributed and robust approach to accelerate Big Data applications storage. Monitoring the overall performance of such a storage setup, along with pattern recognition in gathered statistics, are conducted as inevitable process stages.

Future storage systems will include multiple device types, using different technologies and exhibiting different characteristics. Detecting performance and reliability problems in these systems and properly placing data, emerges as an important problem. Currently, we design a system that is able to monitor a complex storage setup, gather statistics, examine correlations, and generate alerts and possible corrective actions.

More specifically, we examine how system-level metrics can be used to infer application behavior and specifically performance degradation when mixed workloads are deployed on consolidated servers. We develop a lightweight and inexpensive monitoring framework that detects a performance degradation of unknown applications by using only system level metrics.

Gaps in state-of-the-art

High level metrics (HLMs), such as cpu and memory utilization, are the dominant tools to monitor and understand the behavior of systems and applications. This is a well-established approach which every DC operator and administrator has adopted and provides valuable information for the infrastructure behavior. HLMs are gathered from various sources related to the infrastructure, such as hardware counters and operating system statistics.

On the other hand, this approach is unable to provide information about application performance and behavior because typically multiple applications run on servers at any point in time. This is particularly true today with virtualization and the current trend towards workload consolidation.

Therefore, today it is not easy for providers to infer wether an application, treated as a black box, performs in a manner that is desirable or acceptable to the user. However, providers are required to make decisions on application placement (scheduling) as applications arrive, or migration depending on various conditions. Large-scale providers need to schedule incoming applications in a manner that results in acceptable performance for customers without over-provisioning resources. In addition, improved provisioning for applications is important not only for financial reasons but also for infrastructure utilization and energy consumption.

Today, the main way to monitor application performance remains the use of application-level metrics. However, with increasing complexity and diversity in applications it becomes difficult and impractical for providers to have visibility into application metrics. Therefore, an interesting problem today is monitoring the performance and behavior of an application without requiring applications-level metrics.

Goals

- Detect and quantify QoS violation/degradation without requiring application level metrics
- Identify transparently resource requirements for applications
- Identify the root cause of QoS degradation (RCA)

4.3 *ESR15: Root cause analysis system for dynamic large computing environments*

Contributor: *Michał Zasadziński, CA*

Overview

Data storage is the second largest contributor to power consumption, just after CPU [3]. Power-reduction techniques for data center storage system are widely described in [26]. Its energy consumption, can be optimized through different aiding systems and techniques, which are applied both on the software and hardware layer. Software solutions can be focused on file systems, effective caching, data compression and efficient workload oriented data allocation. Besides, hardware layer´s techniques are based on the device design and from the perspective of a data center architecture it means choosing appropriate device types for particular tasks.

Furthermore, the highest maturity level of the data center, which in the current state of the art is seen as visionary (5 years away), fulfills requirements of operational media, e.g., tape, SSD, cloud, HDD choice and an ability to shift storage type based on an energy use policy [34]. The authors of [1,8] defend that the growth of world´s data volume and as a consequence increased energy consumption can be mitigated by more aggressive policies of an effective application of different storage tiers.

The technique of switching between the different storage types is known as a *Hierarchical Storage Management* [29] and more precisely, the object of interest in this PhD research is an automated storage tiering. Research results presented in [19] describe widely a system with a *configuration advisor* for planning and buying storage devices and a *dynamic tier manager* which works with data

replacements between different tiers, to optimize a real time system performance and operating costs. It is important to see, that the energy consumption can be reduced on the stage of hardware choosing for a particular workload as well as through root cause analysis when the system already works. An intensive work about dynamic data allocation was done in [32] and it is not the only example of research on storage tier management during the system runtime. Authors of [19] emphasize that multi-tiered enterprise storage systems represent an important research direction to reduce power consumption and overall operating costs.

Gaps in state-of-the-art

As a consequence of a deep analysis of the state of the art there is a necessity to state the following gaps, which need to be accomplished by proper research. First of all, there is a necessity to provide knowledge and solutions to improve and boost up designing stage of storage system. Newly proposed recommendation systems and policies should be able to assist and point the right storage system structure with an optimal energy consumption and robustness. Improper design results in increase of failures of the storage nodes what should be avoided. Secondly, there is a necessity to perform research considering the cost of a particular storage configuration in dependency with possibility of node failures and their roles in the system.

Goals

- Exploration on how the Root Cause Analysis system could guide the design of the applied storage solutions in data centers for different workloads.
- Use of the accomplished solutions and system to explore an optimal storage structure, in detail by managing the appropriate storage tiering for lower energy consumption.

5 Bibliography

[1] Hormann, P., & Campbell, L. (2014). Data Storage Energy Efficiency in the Zettabyte Era. *Australian Journal of Telecommunications and the Digital Economy , 2* (3).

[2] Adnan, M.A., Sugihara, R., Yan Ma, Gupta, R.K. 2013. Energy-Optimized Dynamic Deferral of Workload for Capacity Provisioning in Data Centers. Green Computing Conference (IGCC). 2013 International. 1 – 10. DOI=10.1109/IGCC.2013.6604515 .

[3] Aroca, J., Chatzipapas, A., Anta, A., & Mancuso, V. (2015). A Measurement-Based Characterization of the Energy Consumption in Data Center Servers. *IEEE Journal on Selected Areas in Communications , 33* (12), 2863-2877.

[4] Bash, C., & Forman , G. (2007). Cool Job Allocation: Measuring the Power Savings of Placing Jobs at Cooling-Efficient Locations in the Data Center. HP Laboratories Palo Alto .

[5] Beloglazov, A., & Rajkumar, B. (2010). *Adaptive Threshold-Based Approach for Energy-Efficient Consolidation of Virtual Machines in Cloud Data Centers.* The University of Melbourne, Australia, CLOUDS Lab, Dept. of Computer Science and Software Engineering, Melbourne.

[6] Beloglazov, A., Buyya, R., Choon Lee, Y., & Zomaya, A. (2011). A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems. In *Advances in computers* (pp. 47-111). Elsevier Inc.

[7] Bennacer, L., Amirat, Y., Chibani, A., Mellouk, A., & Ciavaglia, L. (2015). Self-Diagnosis Technique for Virtual Private Networks Combining Bayesian Networks and Case-Based Reasoning. *IEEE Transactions on Automation Science and Engineering , 12* (1), 354-366.

[8] Bennacer, L., Ciavaglia, L., Ghamri-Doudane, S., Chibani, A., Amirat, Y., & Mellouk, A. (2013). Scalable and fast root cause analysis using inter cluster inference. *IEEE ICC - Next-Generation Networking Symposium*, (pp. 3563-3568).

[9] *BigStorage Official Website*. (n.d.). Retrieved from http://bigstorage.oeg-upm.net

[10] BigStorage Project Proposal for Horizon 2020 H2020-MSCA-ITN-2014 Call.

[11] *BigStorage: Storage-based Convergence between HPC and Cloud to handle Big Data*. (n.d.). (European Comission) Retrieved from http://cordis.europa.eu/project/rcn/193971_en.html

[12] Bronevetsky, G., Laguna, I., Bagchi, S., & R. de Supinski, B. (2012). Automatic fault characterization via abnormality-enhanced classification. *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012).*

[13] Chavira, M., Darwiche, A., & Jaeger, M. (2006). *Compiling relational Bayesian networks for exact inference.* Int. J. Approx. Reasoning 42(1-2).

[14] Chen, Y.-K. (2012). Challenges and Opportunities of Internet of Things. *17th Asia and South Pacific Design Automation Conference*, (pp. 383-388). Sydney.

[15] Darwiche, A. (2003). A Differential Approach to Inference in Bayesian Networks. *Journal of the ACM , 50* (3), 280-305.

[16] Díez, F., & Druzdziel, M. (2006). Canonical probabilistic models for knowledge engineering. Madrid, Spain.

[17] Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999). Learning probabilistic relational models. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, *99*, pp. 1300-1309.

[18] Getoor, L., Friedman, N., Koller, D., Pfeffer, A., & Taskar, B. (2007). Probabilistic Relational Models. In *Introduction to Statistical Relational Learning* (pp. 129-174). MIT Press.

[19] Guerra, J., Pucha, H., Glider, J., Belluomini, W., & Rangaswami, R. (2011). Cost Effective Storage using Extent Based Dynamic Tiering. *In Proceedings of the 9th USENIX conference on File and stroage technologies (FAST'11).* Berkeley, CA, USA: USENIX Association.

[20] Guo, Z., Zhou, D., Lin, H., Yang, M., Long, F., Deng, C., et al. (2011). G2: A Graph Processing System for Diagnosing Distributed Systems. *Proceedings of the 2011 USENIX Annual Technical Conference (USENIX ATC '11).*

[21] Kiciman, E., Maltz, D., & C. Platt, J. (2007). Fast Variational Inference for Large-scale Internet Diagnosis. *Advances in Neural Information Processing Systems 20.*

[22] Kwisthout, J. (2011). Most probable explanations in Bayesian networks: Complexity and tractability. *International Journal of Approximate Reasoning , 52* (9), 1452–1469.

[23] Li, S., Abdelzaher, T., & Yuan, M. (2011). *Tapa: Temperature Aware Power Allocation in Data Center with Map-Reduce.* Proc. Int'l Green Computing Conf. and Workshops (IGCC).

[24] M. Vaquero, L., & Rodero-Merino, L. (2014, October). Finding your Way in the Fog: Towards a Comprehensive Definition of Fog Computing. *ACM SIGCOMM Computer Communication Review , 44* (5), pp. 27-32.

[25] Miyazawa , M., & Nishimura, K. Scalable Root Cause Analysis Assisted by Classified Alarm Information Model Based Algorithm. KDDI R&D Laboratories, Inc., Ohara Fujimino City.

[26] Mullen, S., Bostoen, T., & Berbers, Y. (2013). Power-Reduction Techniques for Data-Center Storage Systems. *ACM Computing Surveys , 45*.

[27] Nguyen, H., Shen, Z., Tan, Y., & Gu, X. (2013). FChain: Toward black-box online fault localization for cloud systems. *Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on*, (pp. 21-30).

[28] Niggemann, O., Biswas, G., S. Kinnebrew, J., Khorasgani, H., Volgmann, S., & Bunte, A. (2015). Data-Driven Monitoring of Cyber-Physical Systems Leveraging on Big Data and the Internet-of-Things for Diagnosis and Control. *Proceedings of the 26th International Workshop on Principles of Diagnosis.* Lemgo.

[29] Ogawa, S., Kamimura, K., Kato, T., Uehara, T., & Okuda, H. (2001). Performance analysis of hierarchical storage management systems for video retrieval system. *Consumer Electronics, 2001. ICCE. International Conference on*, (pp. 328-329).

[30] Ogawa, S., Kamimura, K., Kato, T., Uehara, T., & Okuda, H. (2001). Performance analysis of hierarchical storage management systems for video retrieval system. *Consumer Electronics, 2001. ICCE. International Conference on*, (pp. 328-329).

[31] Pearl, J. (1988). Probabilistic reasoning in intelligent systems: Networks of Plausible Inference. Los Angeles: Morgan Kaufman Publishers.

[32] Shi, H., Arumugam, R. V., Foht, C. H., & Khaing, K. K. (2012). *Optimal disk storage allocation for multi-tier storage system.* Data Storage Institute (DCT Division), Nanyang Technological University.

[33] Shumann, J., Mbaya, T., Mengshoel, O., Pipatsrisawat, K., Srivastava, A., Choi, A., et al. (2013). Software Health Management with Bayesian Networks. *Innovations in Systems and Software Engineering , 9* (4), 271-292.

[34] Singh, H., & Reuters, T. (2011). *Data Center Maturity Model.* The Green Grid.

[35] Wang, C., Schwan, K., Laub, B., Kesavan, M., & Gavrilovska, A. (2014). Exploring Graph Analytics for Cloud Troubleshooting. *11th International Conference on Autonomic Computing.* Philadelphia.

[36] Wuillemin, P., & Torti, L. (2011). *Patterns discovery for efficient structured probabilistic inference.* . In Proceedings of the 5th international conference on Scalable uncertainty management (SUM'11), Salem Benferhat and John Grant (Eds.). Springer-Verlag, Berlin, Heidelberg, 247-260.

[37] Yemini, S., Kliger, S., & Mozes, E. (1996). High Speed and Robust Event Correlation. *IEEE Communications Magazine* , 82-90.

[38] Zermani, S., Dezan, C., Euler, R., & Diguet, J. (2014). Online Inference for Adaptive Diagnosis via Arithmetic Circuit Compilation of Bayesian Networks. *Designing with Uncertainty: Opportunities & Challenges workshop.* York, UK.

[39] Fadika, Z., Dede, E., Hartog, J., & Govindaraju, M. (2012). MARLA: MapReduce for heterogeneous clusters. *Proceedings - 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2012*, 49–56. http://doi.org/10.1109/CCGrid.2012.135

[40] Ananthanarayanan, G., Agarwal, S., Kandula, S., Greenberg, A., Stoica, I., Harlan, D., & Harris, E. (2011). Scarlett : Coping with Skewed Content Popularity in MapReduce Clusters. *Proceedings of the Sixth Conference on Computer Systems - EuroSys '11*, 287. http://doi.org/10.1145/1966445.1966472

[41] Xie, J. X. J., Yin, S. Y. S., Ruan, X. R. X., Ding, Z. D. Z., Tian, Y. T. Y., Majors, J., Qin, X. Q. X. (2010). Improving MapReduce performance through data placement in heterogeneous Hadoop clusters. *Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, *9*, 29–42. http://doi.org/10.1109/IPDPSW.2010.5470880

[42] Knafla. N. *A Prefetching Technique for Object-Oriented Databases*. Advances in Databases (1997). p 154-168.

[43] Wook. S and Kyu, Y. and Y. S. A Formal Framework for Prefetching Based on the Type-Level Access Pattern in Object-Relational DBMSs. IEEE Trans. Knowl. Data Eng. (2005). p 1436-1448

[44] Soundararajan, G. and Mihailescu, M. and Amza, C. *Context-Aware Prefetching at the Storage Server*. USENIX 2008 Annual Technical Conference (2008). p 377-390.

[45] Bernstein, P. and Pal, S. and Shutt, D. *Context-Based Prefetch for Implementing Objects on Relation*s. The VLDB Journal (2000). p 177-189.

[46] Ibrahim, A. and Cook, W. Automatic Prefetching by Traversal Profiling in Object Persistence Architectures. ECOOP (2006).

[47] Ana Klimovic, Christos Kozyrakis, Eno Thereska, Binu John, and Sanjeev Kumar. 2016. Flash storage disaggregation. In Proceedings of the Eleventh European Conference on Computer Systems (EuroSys '16). ACM, New York, NY, USA, , Article 29 , 15 pages. DOI=http://dx.doi.org/10.1145/2901318.2901337

[48] Myoungsoo Jung and Mahmut Kandemir. 2013. Revisiting widely held SSD expectations and rethinking system-level implications. In Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems (SIGMETRICS '13). ACM, New York, NY, USA, 203-216. DOI: http://dx.doi.org/10.1145/2465529.2465548

[49] Vaggelis Atlidakis, Jeremy Andrus, Roxana Geambasu, Dimitris Mitropoulos, and Jason Nieh. 2016. POSIX abstractions in modern operating systems: the old, the new, and the missing. In Proceedings of the Eleventh European Conference on Computer Systems (EuroSys '16). ACM, New York, NY, USA, , Article 19 , 17 pages. DOI=http://dx.doi.org/10.1145/2901318.2901350

[50] Chia-Che Tsai, Bhushan Jain, Nafees Ahmed Abdul, and Donald E. Porter. 2016. A study of modern Linux API usage and compatibility: what to support when you're supporting. In Proceedings of the Eleventh European Conference on Computer Systems (EuroSys '16). ACM, New York, NY, USA, , Article 16 , 16 pages. DOI=http://dx.doi.org/10.1145/2901318.2901341

[51] Tyler Harter, Chris Dragga, Michael Vaughn, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. 2011. A file is not a file: understanding the I/O behavior of Apple desktop applications. In

Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles (SOSP '11). ACM, New York, NY, USA, 71-83. DOI=http://dx.doi.org/10.1145/2043556.2043564

[52] Stan Park and Kai Shen. 2012. FIOS: a fair, efficient flash I/O scheduler. In Proceedings of the 10th USENIX conference on File and Storage Technologies (FAST'12). USENIX Association, Berkeley, CA, USA, 13-13.

[53] Myoungsoo Jung, Wonil Choi, Shekhar Srikantaiah, Joonhyuk Yoo, and Mahmut T. Kandemir. 2014. HIOS: a host interface I/O scheduler for solid state disks. In Proceeding of the 41st annual international symposium on Computer architecuture (ISCA '14). IEEE Press, Piscataway, NJ, USA, 289-300.

[54] Zhang, J., Shu, J., & Lu, Y. (2016). ParaFS: A Log-Structured File System to Exploit the Internal Parallelism of Flash Devices. In 2016 USENIX Annual Technical Conference (USENIX ATC 16). Chicago.